2.0.0 Change Log

Primary Changes

1. Added a new Unity Light Baker Automated Action, which facilitates the baking of lightmaps, reflection probe data, and Light Probe data using the new Automation functionality of the World Designer Tool.

Light Baking can be performed on a cell by cell basis, which allows for partial bakes of the world (versus rebaking one giant scene every time you make a simple change).

2. Modified the way World Regions are stored. Previously, Region Cells were stored with the Index of the Grouping they were associated with. This was not ideal, because if you change the order of Groupings, the Region Cells would become invalidated (i.e., loading the World Region might not load the Cells you expect).

Now Region Cells are stored with the internal ID of the Grouping, which means you can re-order Groupings without issue. Removing a Grouping will break this, however (even if you re-add the Grouping), because the ID of the Grouping will most likely change.

World Regions using the old format will still work at runtime, however it is recommended to "upgrade" the format, which can be done by simply clicking on a World component that has World Regions saved to it (and then saving the scene that World is in), or by opening the World Designer Tool (which will automatically update World Regions stored in the World Designer Data currently in use by the tool).

3. You can now select all Asset Chunks belonging to loaded Streamable Grid Cells via the World Designer Tool (via the Select button in the Editing Controls tab of the Main Controls Menu). You can also deselected selected Asset Chunks belonging to selected Cells using the Deselect button.

4. When Keep Scenes Intact is enabled on a Scene based Chunk Streamer, an error will no longer be thrown if an LOD Group using that Streamer also uses a Hierarchy Organizer. This is to support cases where the scene needs to remain loaded (usually because it contains lighting or other baked data), but the actual Asset Chunk Game Object does not need to remain in that scene.

5. Changed the behavior of the Addressable Prefab Chunk Streamer to make use of the Chunk Destroyer's DestroyChunksOnCells method, which should speed up the destruction process in most situations.

Also added a new setting called Wait On Memory Freeing Op, which allows you to choose whether the Chunk Streamer will wait on the final Memory Freeing operation (if needed) before returning control to the Chunk Manager. Disabling this may speed up the World Update speed in some cases.

6. Added a new setting called Relinquish Control ASAP When Unloading to the Addressable Scene Chunk Streamer. If enabled and if Call Unload Unused Assets is disabled, the Streamer will queue the async unload operations to unload the Asset Chunk scenes but will not wait for those unload operations to finish. Instead, it will relinquish control back to the Chunk Manager immediately. This may speed up the World Update process, however it may also result in more memory pressure and/or unforeseen issues.

If this option is enabled but Call Unload Unused Assets is enabled, the Streamer will unload all scenes, wait for them to be unloaded, then call Resources.UnloadUnusedAssets and immediately relinquish control back to the Chunk Manager.

If this option is disabled, the Streamer waits for the scenes to unload, then will call Resources.UnloadUnusedAssets and wait for it to finish (if Call Unload Unused Assets is enabled).

7. Removed the DestroyChunk method from the ChunkDestroyer class, as it is no longer needed with change 1 mentioned above. This simplifies the API.

8. You can now specify whether Chunk Offsets (found on LOD Groups on Streamable Grid assets) are cell relative or absolute values.

Cell Relative is the previous value for any used axes, where you specify a % (between 0 and 1) that the positions will be offset from the min point on the cell.

Absolute means the value will be added to the Chunk Position directly, which was how the offset for the Unused Axis was specified. So basically, you now have more freedom to use either option for every Axis.

9. Added options for enabling/disabling Behaviours before/after initialization to the SAM Initializer.

10. Changed Hierarchy Based Searches when using the Assignment Operation of the World Designer Tool so that all What To Assign methods can make use of a Regex Pattern (optional unless What To Assign is set to Match Regex Pattern Only).

11. The AutomatedSerializedPropertyModifier Scriptable Asset (part of the Automation System) can now make use of a Regex Pattern to limit the modifications to game objects with specific names.

12. Renamed Print Cells Action to Print Cell Info. This Automated Action can be used to print various information about Streamable Grid Cells and their Asset Chunks, the most useful probably being the ability to print vertex count information about the Cells, their Asset Chunks, and the children of the Asset Chunks.

13. Added the AutomatedChunkCleaner Scriptable Asset to the Automation System. This can currently be used to remove disabled Colliders and Renderers in your Asset Chunks (or their descendants), as well as Mesh Filters with missing meshes and missing scripts. It can also be used to destroy empty (no components other than Transform) childless game objects (but not the root Asset Chunk).

14. Added the AutomatedFindDisabledBehaviours and AutomatedFindDisabledRenderers Scriptable Assets to the Automation System. These can be used to find and print info about disabled components on your Asset Chunks (or their descendants) that inherit from Behaviour or Renderer.

15. Added the AutomatedFindMeshRenderersWithMaterial Scriptable Asset to the Automation System. This can be used to find and print info about Mesh Renderers on Asset Chunks (or their descendants) that contain a material of a specific name.

16. Added the AutomatedNullTreeReplacer Scriptable Asset to the Automation System. This can be used to replace Tree Prototypes that have missing prefabs with a different tree prototype on the same terrain.

17. Added the AutomatedTerrainDetailReplacer Scriptable Asset to the Automation System. This can be used to modify Detail Prototypes that were added as Grass Textures so that they use a prefab instead. You can also alternatively enable GPU instancing for the modified prototypes as well, or change the DetailRenderMode of the prototype.

18. Active Grid Groupings will now be matched to World Groupings based on the Grouping Name assigned to both Groupings. The previous behavior was to perform Index based matching (Active Grid Grouping 1 was matched to World Grouping 1).

The new system offers more flexibility in instances where the Active Grid might sync to multiple Worlds with different Grouping layouts, however you must be careful not to change the Grouping Names if you or your gaming customers have performed a SAM Save operation (via the Component Manager), as the names are stored in the save data now and used to apply Active Grid Grouping related save data (such as changes to Loading Blueprints).

19. Adding the TimingDataTracker component. Add this to a scene and SAM will automatically track various metrics related to asset chunk loading, unloading, activation, and deactivation. This data can be retrieved or automatically printed to the console or some other place. Also added profile markers to the methods these metrics track, which can be used to find out how much frame time is being spent on the different operations.

20. The metrics and profile markers are only available in the editor or in a Development Build for performance reasons!

21. Made some changes to the Automation Process.

22. Saving of cells is now a manual process that you must perform yourself, which you can do by calling one of the methods (SavePrimaryCell, SaveSecondaryCells, SaveTertiaryCells) of the AutomatedActionInput class.

While this may seem like more work than before, this actually gives you greater control over if/when the cells are saved, as you can save the cells, then do more work that you might not want to be saved.

23. The OnAutomateOperationStarting and OnAutomateOperationFinished methods of the AutomatedAction class now take a AutomationOperationInfo object which currently just contains the World the Automate Operation is for (but might contain more info in the future).

24. You can now have the Default Asset Creator automatically add labels to Addressable Asset Chunks corresponding to the Layer, Row, and/or Column of the cell the Asset Chunk is associated with. This is useful for organizing asset chunks belonging to the same cell, for use with the Pack Together By Label packing strategy. There is also a new tool called the Addressable Group Configurer that can be used to add these labels to existing Addressable Asset Chunks (use by creating a Addressable Group Configurer Asset via Assets -> Create -> Deep Space Labs -> SAM -> Addressable Group Configurer).

25. Removed the isApplicationPlaying parameter from the CellVisualTransitionController.ResetToVisibleState and ResetToInvisibleState methods. Use Application.isPlaying instead.

26. Added a one frame delay to the Chunk Manager base class before triggering chunk loading/unloading via a Chunk Streamer. Via testing this showed as much as a 5ms performance boost when loading the first asset chunk in a given world update, although your results may vary, and this boost may be hardly noticeable in some instances.

27. Added a way for code inside the SAM DLL to make use of Unity preprocessor directives. You will find a new asset called ConditionalMembers_DoNotDeleteOrRename inside the Assets/Deep Space Labs/SAM/Resources folder that facilitates this. The asset can be regenerated using the command Assets -> Deep Space Labs -> SAM -> Regenerate Conditional Members Implementation.

At this time, this new system is only being used to add profiling markers to SAM Instantiate and GameObject.SetActive calls, for profiling purposes, though it may be used to add more profiling information or other reasons in the future.

28. Added Zones! Zones can be thought of as discreet areas in your game World where you can have unique content. Each Zone Grouping (intersection of a Zone and World Grouping) makes use of its own unique Streamable Grid, allowing you to create smaller sets of Asset Chunks that can be easier to manage and reason about.

Zones have their own position within a globally defined “World Map” which can be updated at any time, meaning you can move content around if needed. This also makes it easier to expand your game world on the negative axis, as you can simply add new Zones whenever it is necessary to do so.

By default all existing Worlds will be updated automatically to make use of a single Zone, which will effectively equates to the area your game world currently covers. If you have no interest in adding additional Zones, no worries! You don’t have to.

Note, as a result of this change, Streamable Grids are no longer configured with the World Grouping panel of the World editor. Instead, you can set them via the Zones panel. However also note that not all Groupings for a Zone require a Streamable Grid. You may have some Zones which do not need certain types of content (vegetation in a desert, for example), which case you can leave that Grouping’s Streamable Grid field null.

29. Added Chaining, which allows you to chain together different World Groupings so that when you load or unload a World Cell from one Grouping, all Cells that overlap that cell from other Chained Groupings are also loaded or unloaded. You can configure which Groupings are chained in the Operation Settings/Chaining Window/Tab of the World Designer Tool. Chaining is activated when you right click the Load (L*) or Unload (U*) buttons. Any other operations that automatically load/unload cells will not use chaining!

30. Added an Overridable World setting that allows you to activate/deactivate the first level children of root Asset Chunks before the root Asset Chunks when Auto Activate/Deactivate is enabled. This is very useful for Asset Chunks with a ton of children, where before activating/deactivating the root chunk would cause all children to be activated/deactivated in a single frame.

31. Added some new built in methods for organizing Asset Chunks (and their children) when using Multi-Chunking. These can be useful in some specific situations, and this framework also allows you to create custom Multi-Chunk Organizers, which grants you full control over which Chunks of a Cell each child are assigned to, as well as the number of Chunks created. This new organization functionality is used anytime an Evaluate Operation is performed.

32. Removed the restriction of having a fixed number of Chunks per Cell on very large Streamable Grids.

33. Found an unfixable issue with the Extra Scaling setting from the Scale Transitioner, requiring its removal.

34. In the World Designer Tool, replaced the Draw Cells In Scene option (in View Controls) with a new option called Cell Scene Outline. This allows you to specify a border (old drawing method) or a full cuboid transparent mesh for the cell outline shown in the Scene View when one or more World Cells are selected via the World Designer Tool. Transparency is adjustable.

If using Border, an issue with the size of the border mesh was also fixed so it matches the correct size of the Cell.

Finally, some issues with drawing this outline in the scene view were fixed, meaning the outline should now be consistently shown correctly.

35. Added the ability to scale up/down the dimensions of Streamable Grid Cells (available via the Streamable Grid inspector). In most cases, you'll want to halve or double your cells per axis when performing this type of operation!

36. Added a new Global Editor Settings setting which can also be configured from the World Designer Tool's Global Settings window. The new setting called Dirty Check Frequency allows you to control the frequency (in seconds) at which the World Designer Tool will perform it's checks to see if cells/asset chunks are dirty (i.e., have unsaved changes). This is useful when using complex Prefabs as Asset Chunks, as the dirty checks can be quite expensive (you'll get a window that says WorldDesignerTool.Repaint and blocks the editor when this is an issue). You can also set the value to 0 to turn of these automatic checks. Finally, there is a button next to the setting on the World Designer Tool called Force Check that allows you to manually trigger a dirty check.

37. Changed the Game Objects On Layer What To Assign option (World Designer Tool/Assignment) to Game Objects On Layers. Also changed the Layer option to a LayerMask option, allowing you to select multiple Layers that objects can be on to be selected for an Assignment Operation.

38. Added the option to use a tag and layer mask to filter which game objects are modified when using an Automated Serialized Property Modifier Action (World Designer Tool automation system).

39. Added the option to target/modify properties of a sub asset of a component when using an Automated Serialized Property Modifier Action. For instance, this can be used to modify properties of a Serialized Object that might be assigned to a custom Component.

40. When modifying integral type properties using an Automated Serialized Property Modifier Action, you can now choose to add, multiple and divide the value set by the existing property value, in addition to a straight replacement (previously you could just replace or add to the existing value).

41. Added a new Automated Asset Chunk Scaling Action, which allows you to scale Asset Chunks by a set amount uniformly on all axes. This allows you to scale entire scenes up/down as needed, although note that you will need to make modifications to the Streamable Grid Cell Sizes and Zone and/or World positions in order to account for the changed scales.

42. Added an option to specify a Sample Zone Grouping when using the Loading Blueprint Editor, which you can then use to capture certain data. Currently this data includes Streamable Grid rows/columns/layers, as well as loaded cells for the Zone Grouping. Using loaded cell data is a powerful tool which allows you to configure Unique Loading Patterns when using Non Uniform Custom Shape Blueprints.

To use this, first identify the cell in the Blueprint to create a Unique Loading Pattern for, then using the World Designer Tool load that cell, plus any other cells which you think need to be loaded when the player is inside that cell (generally the cells whose content is visible from the cell the player is in). Then, in the Loading Blueprint Editor, inside the Blueprint Data tab, use the “Capture Loaded Cells From Sample Zone Grouping” button at the bottom of the window.

43. The World Designer Tool will now automatically correct incorrect Root States when loading Asset Chunk Scenes (previous behavior was display a popup asking if you want the Root States to be corrected). A warning in the console will be displayed when this occurs.

44. When loading scene based Asset Chunks via the World Designer Tool, if the scenes contain root game objects which are set to not be saved in editor or build, they will be ignored and no error popup will be displayed. These types of objects are commonly created by 3rd party assets.

45. The Game Objects With Tag option in the What To Assign dropdown on the World Designer Tool (Operation Settings/Assignment) has been renamed to Game Objects With Tags. This option now allows you to specify multiple tags to use in the matching. Each tag should be separated by double semicolons (;;).

46. Added Game Start Configurations to the World component. These allow you to setup one or more combinations of a World Origin Cell and Player Position to use at the start of the game, which effectively allow you to control the starting layout of the World’s Cells and thus the starting content of your game. Only one configuration can be active per World, however you can change which configuration is active either using the inspector of the World or using scripting. Configurations can also be setup using the World Designer Tool, which is easier than manually trying to decide which Origin Cell and Player Position to use.

Fixes

1. Fixed a new “Native Collection has not been disposed . . . “ message that was introduced in the last update. It would only occur in instances where the scene SAM was in was unloaded outside of the application quitting.

2. Fixed a bug causing cells to not be updated properly after changing an Active Grid Grouping’s Loading Blueprint at runtime.

3. Fixed a bug causing changing Loading Blueprints in Play Mode via the Active Grid inspector not to work.

4. Fixed an exception that would be thrown when trying to change an Active Grid Grouping's Loading Blueprint to the first Blueprint in a repository while in Play Mode using the dropdown.

5. Fixed a bug that would cause cells to be loaded when changing an Active Grid Grouping's Loading Blueprint when that Grouping has Update World disabled.

6. Fixed Duplicate Data button not working (under Available Data dropdown in World Designer tab of World component inspector).

7. Fixed a bug causing Worlds created at runtime from a prototype with one or more Auto Loaded World Regions not initializing with the Auto Loaded World Region Cells loaded.

8. Fixed a rare race condition that would cause the Active Grid to remain endlessly busy.

9. Fixed a bug causing a warning message (regarding chunk position offsets not being at 0 when using Terrain) to appear incorrectly in the Console Log.

10. Game Objects that would be assigned during an Assignment Operation are now checked to make sure they (or any of their children) do not have missing scripts. This should eliminate the critical error where the World Designer Tool tries to save an Asset Chunk Prefab that has one or more game objects with missing script, but it fails due to Unity not allowing it (which would result in a loss of data).

11. Fixed a bug that would hang the Active Grid when trying to set an Active Grid Grouping to the same Loading Blueprint it is currently using.

12. Fixed a bug that would cause a WorldCellWithTerrain to have a null Terrain within the OnBeforeCellsInBatchDeactivated method of a World Grouping Listener.

13. Fixed an exception that could be thrown when manually removing child objects from a prefab Asset Chunk while using the World Designer Tool.

14. Fixed an exception that could be thrown when using the Transfer Operation of the World Designer Tool. Also fixed an issue that could leave an Asset Chunk incorrectly loaded from a Transfer From Cell.

15. The World will now better catch instances when multiple LOD Groups across different World Groupings use the same Group Name, and will stop you from launching

the World Designer Tool or editing an LOD Group when it detects this. This is important as using the same Group name is not allowed and can cause issues with SAM.

16. Fixed a bug causing the Load In Deactivated State setting to not work properly on the Addressable Prefab Chunk Streamer.

17. Fixed a bug causing Terrain Neighboring to not work properly when using Two Frame Initialization.

18. Fixed an ArgumentException that could occur using a Scene Chunk Streamer without a Chunk Destroyer, when Keep Scenes Intact enabled and when using a Hierarchy Organizer.

19. Fixed a bug with the callback logic when using World. RegisterOnUserRequestsFulfilledCallback that could result in a callback not being triggered at the appropriate time.

20. TryGetOriginCellOfGrouping (on AutomatedActionInput class) has been fixed to return a 1 based Origin Cell value rather than a 0 based Origin Cell value.

21. Fixed an IndexOutOfRangeException that could occur when using World Regions on a World component.

22. Fixed an issue with using the “All” button on an Active Grid Grouping to set all Groupings to use the same blueprint as the current Grouping. This issue would result in a warning message repeatedly being displayed when selecting the Active Grid.

23. Fixed Automate Operation Incorrect Progress tracking.

24. Fixed a NullReferenceException in the SceneChunkStreamer that occurred when Keep Scenes Intact was disabled.

25. Fixed an IndexOutOfRangeException in the DefaultSceneChunkStreamerAssetManager class when using the Assignment, Transfer, and Evaluate Operations.

26. Fixed window tabs in custom SAM Editor Windows/Inspectors reverting to all inactive, making it hard to know which tab/window is active.

27. Fixed an IndexOutOfRange Exception that could occur in some rare instances when using the World Designer Tool.

28. Fixed a NullReferenceException when trying to retrieve a scene or prefab addressable asset via the World Designer Tool, when an asset with the same name but of a different type is also present in Addressable Groups.

29. Many other undocumented fixes.

30. Fixed a NullReferenceException that would occur when clicking on a World component that was just added to a scene.

31. Fixed changing Time.timeScale affecting loading in most scenarios. Cell Visual Transition Controllers can still be affected by disabling their new setting called Ignore Time Scale. Components which make use of FixedUpdate (such as Player Movers and the Standard Hierarchy World Shifter) can still be affected!

32. Fixed an issue with Transfer Operations using the World Designer Tool when transferring objects from one LOD Group to another on the same Zone and Grouping. The issue could result in the cell being disabled and all Asset Chunks destroyed, rather than the expected behavior of the objects being transferred to the Asset Chunks of the target LOD, and removed from the source LOD.

33. Fixed View Controls Start/End Row/Column/Layer values when using the World Designer Tool and Loading Blueprint Editor, to allow you to set the Start value without affecting the End Value, and vice versa. This fixes an issue where if you were changing the End Value, SAM would temporarily think the End Value was smaller than you intended (i.e., setting 92, when you enter the 9, it treats the End Value as 9 temporarily), which would result in the Start Value being changed to the first digit, since the Start Value cannot be larger than the End Value.

Now, you can configure the values as you need, and they will only be restricted when you press the “Update View Area” button.

34. Fixed an issue with Transfer Operations using the World Designer Tool when transferring objects from one LOD Group to another on the same Zone and Grouping. The issue could result in the cell being disabled and all Asset Chunks destroyed, rather than the expected behavior of the objects being transferred to the Asset Chunks of the target LOD, and removed from the source LOD.

35. Fixed View Controls Start/End Row/Column/Layer values when using the World Designer Tool and Loading Blueprint Editor, to allow you to set the Start value without affecting the End Value, and vice versa. This fixes an issue where if you were changing the End Value, SAM would temporarily think the End Value was smaller than you

intended (i.e., setting 92, when you enter the 9, it treats the End Value as 9 temporarily), which would result in the Start Value being changed to the first digit, since the Start Value cannot be larger than the End Value.

Now, you can configure the values as you need, and they will only be restricted when you press the “Update View Area” button.

Other

1. You can now Start Editing a World directly from a World Designer Tool window that is not current being used to edit any Worlds. This will make it easier to restart the design/edit process if the World becomes desynced from the Tool (usually due to an assembly reload).

2. Changed the behavior of runtime Component Manager methods that allow you to create Worlds and Active Grids, so that an exception will be thrown if you call the methods before the Component Manager has been fully initialized. Since you can only use these methods after the Component Manager has been initialized, this also means that when creating Worlds, any initial cells on that World will be loaded in a gradual manner. Use RegisterOnUserRequestsFulfilledCallback to get notified when initial cells (and their asset chunks) are fully loaded.

3. Online Docs/API

4. Clarified/Revised public API on existing Position Translation methods on the World class.

5. Added a new method to the World component (GetGroupingName) allowing you to retrieve the name of each Grouping as defined by you in the inspector.

6. Added a new method (GetEndlessGridCellsInAxisAlignedBoundingCube) to the World class, which can be used to get all endless grid cells that fall within an axis aligned bounding cube within the World.

7. Renamed the Addressable Prefab Chunk Streamer’s Wait On Memory Freeing Op option to Wait On Final Memory Freeing Op. Also added this setting to the Prefab Chunk Streamer and Scene Chunk Streamer.

8. Added four new events to the Component Manager class, allowing you to be notified just before and after a World or Active Grid has been initialized. These events are ActiveGridInitializing, ActiveGridInitialized, WorldInitializing, and WorldInitialized.

9. When using the Assign At Runtime Player option on an Active Grid, and you have assigned a TransformPlayer or CustomPlayer, a button will be displayed at runtime allowing you to select the Player game object that is assigned to the Active Grid.